Prime Time Algorithms!

Constructing a Prime Number Generator

Sarah E. Ritchey

Youngstown State University

April 24, 2013

Prime Number Facts

Prime Number Facts

1. Euclid showed that there are infinitely many prime numbers.

Prime Number Facts

1. Euclid showed that there are infinitely many prime numbers.
2. Largest known prime number is $2^{57,885,161} - 1$. It has 17,425,170 digits.

Prime Number Facts

1. Euclid showed that there are infinitely many prime numbers.
2. Largest known prime number is $2^{57,885,161} - 1$. It has 17,425,170 digits.
3. Smallest prime number is 2, and is the only even prime.

Prime Number Facts

1. Euclid showed that there are infinitely many prime numbers.
2. Largest known prime number is $2^{57,885,161} - 1$. It has 17,425,170 digits.
3. Smallest prime number is 2, and is the only even prime.
4. Determining if a large number is prime is computationally feasible.

Prime Number Facts

1. Euclid showed that there are infinitely many prime numbers.
2. Largest known prime number is $2^{57,885,161} - 1$. It has 17,425,170 digits.
3. Smallest prime number is 2, and is the only even prime.
4. Determining if a large number is prime is computationally feasible.
5. Factoring a large number into its prime components is computationally intractable.

Polynomial Prime Generators

Consider the polynomial (with domain restricted to integers)

$$f(n) = n^2 + n + 41.$$

Polynomial Prime Generators

Consider the polynomial (with domain restricted to integers)

$$f(n) = n^2 + n + 41.$$

Here are some facts about this function.

- This function produces only prime numbers for $n = 0, 1, ..., 39$.

Polynomial Prime Generators

Consider the polynomial (with domain restricted to integers)

$$f(n) = n^2 + n + 41.$$

Here are some facts about this function.

- This function produces only prime numbers for $n = 0, 1, ..., 39$.
- But, $f(40) = 40^2 + 40 + 41 = 40 \cdot 41 + 41 = 41^2$.
- For the first 100 inputs, 86 are prime.

Polynomial Prime Generators

Consider the polynomial (with domain restricted to integers)

$$f(n) = n^2 + n + 41.$$

Here are some facts about this function.

- This function produces only prime numbers for $n = 0, 1, ..., 39$.
- But, $f(40) = 40^2 + 40 + 41 = 40 \cdot 41 + 41 = 41^2$.
- For the first 100 inputs, 86 are prime.
- For $0 \leq n \leq 10^6$, f generates 261,081 primes.

Polynomial Prime Generators

Consider the polynomial (with domain restricted to integers)

$$f(n) = n^2 + n + 41.$$

Here are some facts about this function.

- This function produces only prime numbers for $n = 0, 1, ..., 39$.
- But, $f(40) = 40^2 + 40 + 41 = 40 \cdot 41 + 41 = 41^2$.
- For the first 100 inputs, 86 are prime.
- For $0 \leq n \leq 10^6$, f generates 261,081 primes.

Not bad, but not all primes!
Can we construct a polynomial whose outputs are always prime, for integer inputs?

No Polynomial Prime Generator Exists

Consider the polynomial

$$f(x) = a_k x^k + a_{k-1} x^{k-1} + \cdots + a_1 x + a_0.$$

No Polynomial Prime Generator Exists

Consider the polynomial

$$f(x) = a_k x^k + a_{k-1} x^{k-1} + \cdots + a_1 x + a_0.$$

Suppose that $f(n_0) = p$, where n_0 is an integer and p is prime. Let t be an integer. Now,

$$\begin{aligned} f(n_0 + tp) &= a_k (n_0 + tp)^k + \cdots + a_1 (n_0 + tp) + a_0 \\ &= (a_k n_0^k + a_{k-1} n_0^{k-1} + \cdots + a_1 n_0 + a_0) + pQ(t) \\ &= f(n_0) + pQ(t) \\ &= p + pQ(t) \\ &= p(1 + Q(t)) \end{aligned}$$

No Polynomial Prime Generator Exists

Consider the polynomial

$$f(x) = a_k x^k + a_{k-1} x^{k-1} + \cdots + a_1 x + a_0.$$

Suppose that $f(n_0) = p$, where n_0 is an integer and p is prime. Let t be an integer. Now,

$$\begin{aligned} f(n_0 + tp) &= a_k(n_0 + tp)^k + \cdots + a_1(n_0 + tp) + a_0 \\ &= (a_k n_0^k + a_{k-1} n_0^{k-1} + \cdots + a_1 n_0 + a_0) + pQ(t) \\ &= f(n_0) + pQ(t) \\ &= p + pQ(t) \\ &= p(1 + Q(t)) \end{aligned}$$

- We conclude that $p | f(n_o + tp)$.

No Polynomial Prime Generator Exists

Consider the polynomial

$$f(x) = a_k x^k + a_{k-1} x^{k-1} + \cdots + a_1 x + a_0.$$

Suppose that $f(n_0) = p$, where n_0 is an integer and p is prime. Let t be an integer. Now,

$$\begin{aligned} f(n_0 + tp) &= a_k (n_0 + tp)^k + \cdots + a_1 (n_0 + tp) + a_0 \\ &= (a_k n_0^k + a_{k-1} n_0^{k-1} + \cdots + a_1 n_0 + a_0) + pQ(t) \\ &= f(n_0) + pQ(t) \\ &= p + pQ(t) \\ &= p(1 + Q(t)) \end{aligned}$$

- We conclude that $p | f(n_o + tp)$.
- But by assumption, $f(n_o + tp) = p$ for all t.
- This can only occur no more than k times.

No Polynomial Prime Generator Exists

Consider the polynomial

$$f(x) = a_k x^k + a_{k-1} x^{k-1} + \cdots + a_1 x + a_0.$$

Suppose that $f(n_0) = p$, where n_0 is an integer and p is prime. Let t be an integer. Now,

$$\begin{aligned} f(n_0 + tp) &= a_k(n_0 + tp)^k + \cdots + a_1(n_0 + tp) + a_0 \\ &= (a_k n_0^k + a_{k-1} n_0^{k-1} + \cdots + a_1 n_0 + a_0) + pQ(t) \\ &= f(n_0) + pQ(t) \\ &= p + pQ(t) \\ &= p(1 + Q(t)) \end{aligned}$$

- We conclude that $p | f(n_o + tp)$.
- But by assumption, $f(n_o + tp) = p$ for all t.
- This can only occur no more than k times.
- Thus, such a polynomial can not be constructed.

Prime Generation by Ancient Chinese

- Over 25 centuries ago, the Chinese believed that

$$n \text{ is prime if and only if } n | 2^n - 2$$

- For example: $5 | 2^5 - 2 = 30$

Prime Generation by Ancient Chinese

- Over 25 centuries ago, the Chinese believed that

$$n \text{ is prime if and only if } n | 2^n - 2$$

- For example: $5 | 2^5 - 2 = 30$
- Interestingly, this conjecture holds for the first 340 natural numbers.
- But, $341 | 2^{341} - 2$ even though $341 = 11 \cdot 31$

Prime Generation by Ancient Chinese

- Over 25 centuries ago, the Chinese believed that

$$n \text{ is prime if and only if } n | 2^n - 2$$

- For example: $5|2^5 - 2 = 30$
- Interestingly, this conjecture holds for the first 340 natural numbers.
- But, $341|2^{341} - 2$ even though $341 = 11 \cdot 31$
- In fact, it has been proven that there are infinitely many such “pseudoprimes.”
- Lots and lots of research has been done with these types of numbers.

Prime Generation by Ancient Chinese

- Over 25 centuries ago, the Chinese believed that

$$n \text{ is prime if and only if } n \mid 2^n - 2$$

- For example: $5 \mid 2^5 - 2 = 30$
- Interestingly, this conjecture holds for the first 340 natural numbers.
- But, $341 \mid 2^{341} - 2$ even though $341 = 11 \cdot 31$
- In fact, it has been proven that there are infinitely many such “pseudoprimes.”
- Lots and lots of research has been done with these types of numbers.
- Must admit that for the calculation power available that long ago, the Chinese had a great formula.

Other Prime Generation

These formulas are proven to generate primes

-

$$g(n) = \sum_{i=1}^{n-1} \left[\frac{\left[\frac{n}{i} \right]}{\frac{n}{i}} \right] \text{ when } g(n) = \begin{cases} 1, & \text{n is prime} \\ > 1, & \text{n is composite} \end{cases}$$

Other Prime Generation

These formulas are proven to generate primes

- $$g(n) = \sum_{i=1}^{n-1} \left[\frac{\left[\frac{n}{i}\right]}{\frac{n}{i}} \right] \text{ when } g(n) = \begin{cases} 1, & \text{n is prime} \\ > 1, & \text{n is composite} \end{cases}$$

- Primes can also be generated recursively by letting

$$a_n = a_{n-1} + \gcd(n, a_{n-1}), \text{ and } a_1 = 7.$$

Then, the sequence of differences $a_{n+1} - a_n$, $1, 1, 1, 5, 3, 1, 1, 1, 1, 11, 3, 1, 1, ...$, contains only ones and primes.

W. H. Mills

In 1947, W. H. Mills proved the following theorem.

Theorem

There exists a constant A such that $\left[A^{3^n}\right]$ *is a prime for every positive integer n.*

W. H. Mills

In 1947, W. H. Mills proved the following theorem.

Theorem
There exists a constant A such that $\left[A^{3^n}\right]$ *is a prime for every positive integer n.*

The proof is remarkably short.

Important Lemmas

Let p_n denote the nth prime number. A.E. Ingham has shown that

$$p_{n+1} - p_n < K p_n^{\frac{5}{8}}$$

where K is a fixed positive integer.

Important Lemmas

Let p_n denote the nth prime number. A.E. Ingham has shown that

$$p_{n+1} - p_n < Kp_n^{\frac{5}{8}}$$

where K is a fixed positive integer.

Lemma

If N is an integer greater then K^8 there exists a prime p such that $N^3 < p < (N+1)^3 - 1$.

Important Lemmas

Let p_n denote the nth prime number. A.E. Ingham has shown that

$$p_{n+1} - p_n < Kp_n^{\frac{5}{8}}$$

where K is a fixed positive integer.

Lemma

If N is an integer greater then K^8 there exists a prime p such that $N^3 < p < (N+1)^3 - 1$.

Proof.

Let p_n be the greatest prime less then N^3. Then

$$N^3 < p_{n+1} < p_n + Kp_n^{\frac{5}{8}} < N^3 + KN^{\frac{15}{8}} < N^3 + N^2 < (N+1)^3 - 1$$

□

Proof of Mills' theorem

Let P_0 be a prime greater then K^8. Then by the lemma we can construct an infinite sequence of primes, $P_0, P_1, P_2, \ldots,$ such that $P_n^3 < P_{n+1} < (P_n + 1)^3 - 1$.

Proof of Mills' theorem

Let P_0 be a prime greater then K^8. Then by the lemma we can construct an infinite sequence of primes, $P_0, P_1, P_2, \ldots,$ such that $P_n^3 < P_{n+1} < (P_n + 1)^3 - 1$. Let

$$u_n = P_n^{3^{-n}}, v_n = (P_n + 1)^{3^{-n}}$$

Then $v_n > u_n$, so,

$$u_{n+1} = P_{n+1}^{3^{-n-1}} > P_n^{3^{-n}} = u_n \qquad (1)$$

$$v_{n+1} = (P_{n+1} + 1)^{3^{-n-1}} < (P_n + 1)^{3^{-n}} = v_n \qquad (2)$$

Proof of Mills' theorem Continued

Then, $\{u_n\}$ is bounded and monotone increasing.

Thus by Uniform Convergence Theorem, the sequence converges.

Proof of Mills' theorem Continued

Then, $\{u_n\}$ is bounded and monotone increasing.

Thus by Uniform Convergence Theorem, the sequence converges.

Let

$$A = \lim_{n \to \infty} u_n.$$

From (1) and (2), it follows that

$$u_n < A < v_n,$$

or

$$P_n < A^{3^n} < P_n + 1.$$

Proof of Mills' theorem Continued

Then, $\{u_n\}$ is bounded and monotone increasing.

Thus by Uniform Convergence Theorem, the sequence converges.

Let

$$A = \lim_{n\to\infty} u_n.$$

From (1) and (2), it follows that

$$u_n < A < v_n,$$

or

$$P_n < A^{3^n} < P_n + 1.$$

Therefore $\left[A^{3^n}\right] = P_n$, and $\left[A^{3^n}\right]$ is a prime generating function.

The Smallest A

There are many values of *A* that would generate primes. In 2005, Caldwell and Cheng calculated that the minimum Mills' constant (for the exponent c=3) begins with the following 600 digits:

1.3063778838	6308069046	8614492602	6057129167	8458515671
3644368053	7599664340	5376682659	8821501403	7011973957
0729696093	8103086882	2388614478	1635348688	7133922146
1943534578	7110033188	1405093575	3558319326	4801721383
2361522359	0622186016	1085667905	7215197976	0951619929
5279707992	5631721527	8412371307	6584911245	6317518426
3310565215	3513186684	1550790793	7238592335	2208421842
0405320517	6890260257	9344300869	5290636205	6989687262
1227499787	6664385157	6619143877	2844982077	5905648255
6091500412	3788524793	6260880466	8815406437	4425340131
0736114409	4137650364	3793012676	7211713103	0265228386
6154666880	4874760951	4410790754	0698417260	3473107746

Since then, the first 10,000 digits have been calculated and are available at OEIS website!

How Much Accuracy is Needed?

We calculate $\left[A^{3^n}\right]$ up to $n = 19$ using different degrees of accuracy for Mill's constant. Then, we tested to see if they were prime using a probabilistic algorithm for the large numbers and a deterministic algorithm for the small ones.

How Much Accuracy is Needed?

We calculate $\left[A^{3^n}\right]$ up to $n = 19$ using different degrees of accuracy for Mill's constant. Then, we tested to see if they were prime using a probabilistic algorithm for the large numbers and a deterministic algorithm for the small ones.

- For 10 digits of A, we only get primes for $n = \{1, 2\}$.

How Much Accuracy is Needed?

We calculate $\left[A^{3^n}\right]$ up to $n = 19$ using different degrees of accuracy for Mill's constant. Then, we tested to see if they were prime using a probabilistic algorithm for the large numbers and a deterministic algorithm for the small ones.

- For 10 digits of A, we only get primes for $n = \{1, 2\}$.
- For 100 digits of A, we get primes for $n = \{1, 2, 3, 4, 5\}$.

How Much Accuracy is Needed?

We calculate $\left[A^{3^n}\right]$ up to $n = 19$ using different degrees of accuracy for Mill's constant. Then, we tested to see if they were prime using a probabilistic algorithm for the large numbers and a deterministic algorithm for the small ones.

- For 10 digits of A, we only get primes for $n = \{1, 2\}$.
- For 100 digits of A, we get primes for $n = \{1, 2, 3, 4, 5\}$.
- For 1,000 digits of A, we get primes for $n = \{1, 2, 3, 4, 5, 6, 7\}$.

How Much Accuracy is Needed?

We calculate $\left[A^{3^n}\right]$ up to $n = 19$ using different degrees of accuracy for Mill's constant. Then, we tested to see if they were prime using a probabilistic algorithm for the large numbers and a deterministic algorithm for the small ones.

- For 10 digits of A, we only get primes for $n = \{1, 2\}$.
- For 100 digits of A, we get primes for $n = \{1, 2, 3, 4, 5\}$.
- For 1,000 digits of A, we get primes for $n = \{1, 2, 3, 4, 5, 6, 7\}$.
- For 10,000 digits of A, we only get primes for $n = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

How Much Accuracy is Needed?

We calculate $\left[A^{3^n}\right]$ up to $n = 19$ using different degrees of accuracy for Mill's constant. Then, we tested to see if they were prime using a probabilistic algorithm for the large numbers and a deterministic algorithm for the small ones.

- For 10 digits of A, we only get primes for $n = \{1, 2\}$.
- For 100 digits of A, we get primes for $n = \{1, 2, 3, 4, 5\}$.
- For 1,000 digits of A, we get primes for $n = \{1, 2, 3, 4, 5, 6, 7\}$.
- For 10,000 digits of A, we only get primes for $n = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

This is not as surprising as it sounds.

$\left[A^{3^{10}}\right]$ is actually about $23,000$ digits long!

Also, It takes about a gigabyte of memory to store $\left[A^{3^{20}}\right]$.

Conclusions

- When producing primes we are left with a quandary.
- The use of polynomials and other standard function produce primes only some of the time.
- Although, Mills' Function produces only primes, the exponential nature of the function quickly exceeds current computing power and is really not practical given that we were only able to calculate nine primes.
- This implies that there is much work left to do in this area.

Future Research

- I would like to learn how to calculate more digits of Mills' constant.
- I would also like to expand my programing knowledge to be able to work with even larger numbers with more precision.

Thanks to Dr. Ritchey for excellent advisement
and
Tim Shaffer for programing help!

Bibliography

1. W. H. Mills. *A Prime Representing Function*. Princeton University.
2. Caldwell and Cheng. *Determining Mill's Constant and a Note on Honaker's Problem*. Journal of Integer Sequences.
3. The On-Line Encyclopedia of Integer Sequences. http://oeis.org/
4. Burton, David. *Elementary Number Theory*. Seventh Edition.

Thanks to Dr. Ritchey for excellent advisement and Tim Shaffer for programing help!